

RiMOM2013 Results for OAEI 2013

Qian Zheng¹, Chao Shao¹, Juanzi Li¹, Zhichun Wang² and Linmei Hu¹

¹ Tsinghua University, China {zy, shaochao, ljz}@keg.tsinghua.edu.cn

² Beijing Normal University, Beijing, China zcwang@bnu.edu.cn

Abstract. This paper presents the results of RiMOM2013 in the Ontology Alignment Evaluation Initiative (OAEI) 2013. We participated in three tracks of the tasks: Benchmark, IM@OAEI2013, and Multifarm. We first describe the basic framework of our matching System (RiMOM2013); then we describe the alignment process and alignment strategies of RiMOM2013, and then we present specific techniques used for different tracks. At last we give some comments on our results and discuss some future work on RiMOM2013.

1 Presentation of the system

Recently, ontology is increasingly seen as an apocalyptic factor for enabling interoperability between heterogeneous systems and Semantic Web applications. Ontology Aligning is required for combining distributed and motley ontologies. Developing ontology alignment systems has become an essential issue of recent ontology research.

RiMOM2013 is named after RiMOM(Risk Minimization based Ontology Mapping) which is a multi-strategy ontology alignment system and was firstly developed in 2007 [1][2]. RiMOM implements several different matching strategies that have been defined based on different ontological information. For different ontology mapping tasks, RiMOM can automatically select and combine multiple strategies to generate accurate alignment results. RiMOM has evolved all the time since 2007, and RiMOM2013 is developed based on RiMOM and has several new characteristics that will be described in following subsections.

1.1 State, purpose, general statement

As shown in Fig. 1, the whole system is consists of three layers: *User Interface layer, Control layer and Component layer*. In the *User Interface layer*, RiMOM2013 provides an interface to allow customizing the matching procedure: including selecting preferred components, setting the parameters for the system, choosing to use translator tool or not. In semi-automatic ontology matching, the task layer stores parameters of the alignment tasks, and controls the execution process of components in the component layer. In component layer, we define six groups of executable components, including preprocessor, matcher, aggregator, evaluator, postprocessor and other utilities. In each group, there are several instantiated components. For a certain alignment task, user can select appropriate components and execute them in desired sequence.

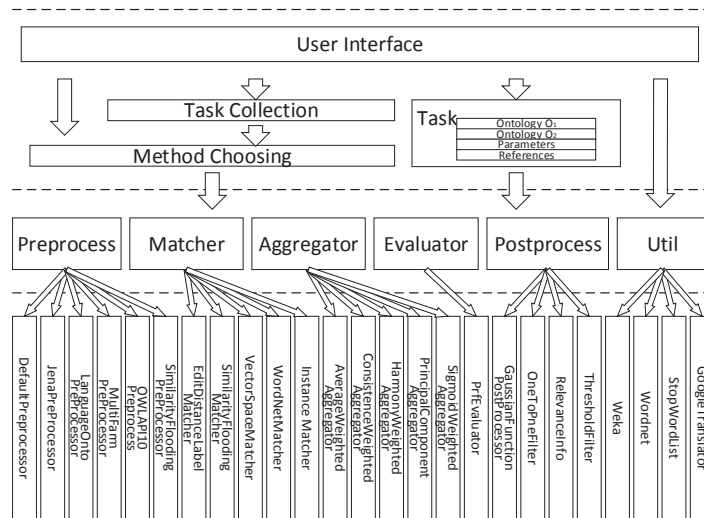


Fig. 1. Framework of RiMOM2013

1.2 Specific techniques used

This year we participate in three tracks of the campaign: Benchmark, Multifarm, and Instance Matching. We will describe specific techniques used in different tracks as follows:

Benchmark

For benchmark track, we use five matching methods: Similarity preprocessor, Similarity matcher, Similarity Flooding preprocessor, Similarity Flooding matcher, and Similarity Aggregator.

We use Edit Distance method and WordNet 2.0 to calculate the similarity between labels of entities, then for each entity pair we combine these two similarities to an aggregated similarity.

Experiments are did on five different flooding methods based on similarity flooding[3]: Property Only Method(POM), Hierarchy Method(HM), Common Relation Method(CRM), RDFGraphOfSchema Method(RGSM) and Nothing Method(NM). These five methods are used to generate the initial graph only for the next step. In POM we add entity pairs which have superclass relationship; And in HM, we add entity pairs which have subclass and super property; In CRM, first we check the relationship between each two entities, then we add entity pairs which have domain relationship or range relationship. In RGSM, we add these pairs either contented in HM and CRM. And for NM, we add all entity pairs into initialize graph.

In the next two steps we use the similarity flooding method to flood the similarities in the graph, and because the map is usually gargantuan, we use a threshold filter to prune the pairs whose similarity smaller than threshold when after the flooding process.

Next we use Aggregator to combine these similarities: EditDistance similarity, WordNet 2.0 similarity, similarity Flooding result similarity. The experiment reflects that the

only single task list without aggregator and other similarities(EditDistance and WordNet 2.0) gains the best result.

Multifarm

The multifarm track is designed to test the aligning systems' ability on multi-lingual dataset[4]. The multifarm data is composed of a set of ontologies translated in seven different languages and the corresponding alignments between these ontologies. Each entity in one ontology requested to be matched with related entity in different language ontology.

The nodus makes this task difficult is that there is restricted information in each entity, which usually only has label information like "writes contribution", and the label of its range property of this entity is "contribution", the label of its domain property of this entity is "author", which when translated into same language usually got same or almost same result like "autor" in Spanish.

In the first preprocess step in multifarm task, we use google translate tool to make two different language into same language, such as when we do the "en-cn" alignment task, we translate the Chinese label to English, and when we do the "cn-es" alignment task, we translate Spanish label into Chinese. Particularly, when either the source ontology or target ontology's language is Russian, we translate them both into English.

In the second preprocess step, we use google translate tool to make two different entities's label all in English for the purpose of use wordnet 2.0 in order to calculate the sentence similarity.

Next we use Aggregator to combine these two similarities for each label pair, the experiment reflects that the edit-distance contributes more in the combined-similarity.

Instance Matching

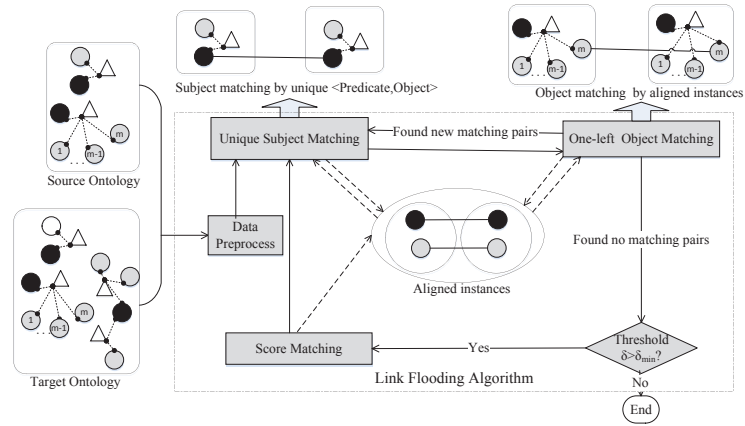


Fig. 2. Framework of instance matching system

For instance matching task, we propose an algorithm called Link Flooding Algorithm inspired by [5], which includes four main modules, namely Data Preprocess, Unique

Subject Matching, One-left Object Matching and Score Matching. Before going into the details, we first define ontology *Ont* as a set of RDF triples $\langle s, p, o \rangle$ (Subject, Predicate, Object), and instance *Ins* as a set of many RDF triples having the same Subject. Since an instance's subject could be another's object, we consider instance matching in three situations: subject and subject alignment, subject and object alignment and object and object alignment.

In the first module called data preprocess, we purify the data including transferring the data sets which are multilingual to be uniform in English. Additionally, we unify the format of data, for example, the date expressed as "august, 01, 2013" or "August, 01, 2013" is transformed to "08, 01, 2013". We also do a lot of other operations like removing special characters to clean the data. The second module achieves instance matching through one unique $\langle p, o \rangle$ for the two instances to be aligned. For example, if in source ontology, only one instance, INS_X has $\langle p, o \rangle$ as $\langle birthday, "01, 08, 2013" \rangle$, then in target ontology, instances containing $\langle birthday, "01, 08, 2013" \rangle$ are concluded to be aligned with INS_X . Consequently, one instance in source ontology can be matched with arbitrary number of instances in target ontology. In the third module, we obtain object and object alignment via all of the aligned subjects. In detail, if two aligned instances have a same predicate both having m objects, of which $m - 1$ are aligned, then the "one-left" object is aligned. The last module is named Score Matching where we consider two instances aligned if the weighted average score of their comments, mottos, birthDates, and almaMaters is above a certain threshold. In this task, we take the edit distance as score measure of similarity. We illustrate the algorithm in Fig. 2.

We first input source ontology and target ontology into the algorithm, as shown in the picture, the black circles represent the subjects of the RDF triples, the gray circles represent the objects of the RDF triples, and the white triangles represent the predicate[6]. We then clean the data set with the module Data Preprocess. Next, we generate some initial instance matching pairs as seeds through Unique Subject Matching. As we mentioned previously, one instance's subject could be another's object, we can input the seeds to One-left Object Matching to get more matching pairs. With those new detected matching pairs, we reapply Unique Subject Matching to acquire more new matching pairs. So, we can iteratively run these two modules until we can not find any new matching pairs. After that, we need to run the Score Matching module with a high threshold to get new pairs with high confidence, thus we can repeat previous operation namely iteratively running Unique Subject Matching and One-left Object Matching module with little error. Later, we reduce the threshold step by step, where in each step newfound pairs are input into the repeated previous operation to control error propagation. Lastly, we output all of the matching pairs if the threshold is below the minimum threshold or all the instances in target ontology are aligned.

1.3 Adaptations made for the evaluation

Deploying the system on seals platform by a network bears three main challenges. Firstly, the input source can not download as a file, we can hardly see the information and structure inherently. Secondly, without the input string path, we can not determine which task and which size of the dataset are now using. Lastly, with the calling the

interface we provide by seals platform, some XML reader problems occur and make the process interrupt, then we have no choice but to discard the XML read and load component to make the system executable, but in multifarm task, we found that there is some difference between the result generated by our local pc and by seals platform, there may have some undiscoverable problems when we turn RiMOM2013 as a unvarying-purpose system.

1.4 Link to the system and parameters file

The RiMOM2013 system can be found at

<http://keg.cs.tsinghua.edu.cn/project/RiMOM/>

2 Results

As introduced before, RiMOM2013 participates in three tracks in OAEI 2013. In the following section, we present the results and related analysis for the individual OAEI 2013 tracks below.

2.1 benchmark

There are two test set this year, biblio and finance, and for each dataset there are 94 align tasks. We divide these tasks into four groups, 101, 20x, 221-247 and 248-266. We got good result on 221-247 and the result turns bad on 248-266, compared with the 2010's result, the evaluate fashion is changed this year, and there is some error during the system docking mission, when we try to use a XML loader to implement circuit-customize, the incompatible problem occurred and because of we do not know the exactly version of the tool seals platform called, we have to write the program imitation separately and make them inflexible. As RiMOM2013 is an dynamic system, these problem more or less affected our implementation.

| DataSet | Precision | Recall | F1-measure |
|---------|-----------|--------|------------|
| 101 | 0.84 | 1.00 | 0.91 |
| 20x | 0.57 | 0.52 | 0.53 |
| 221-247 | 0.71 | 1.00 | 0.82 |
| 248-266 | 0.46 | 0.48 | 0.45 |

Table 1. Benchmark Result of biblio-dataset

2.2 multifarm

There are 36 language pairs in multifarm data set, these pairs is combined with 8 languages: Chinese(cn), Czech(cz), Dutch(nl), French(fr), German(de), Portuguese(pt), Russian(ru), Spanish(es). And permutate depend on lexicographical order. Results are show in Table. 1.

Result is shown in Table 2 and this result is from OAEI2013 result page. It is notable that our system got the minimum runtime among the multilingual matchers, which is not put in this table. Although we got the third rank in multifarm task, we still have to mention that our system basically is a translation based system and the connection with the translator’s supplier is not that good. Otherwise, we could have made it much better. We have proven it locally with no edas and ekaw ontologies, getting F1 as 0.49.

| Language Pair | F1-measure | Language Pair | F1-measure | Language Pair | F1-measure |
|---------------|------------|---------------|------------|---------------|------------|
| cn-cz | 0.120 | cz-nl | 0.320 | en-pt | 0.360 |
| cn-de | 0.180 | cz-pt | 0.240 | en-ru | NaN |
| cn-en | 0.250 | cz-ru | NaN | es-fr | 0.360 |
| cn-es | 0.170 | de-en | 0.390 | es-nl | 0.290 |
| cn-fr | 0.170 | de-es | 0.310 | es-pt | 0.400 |
| cn-nl | 0.160 | de-fr | 0.290 | es-ru | NaN |
| cn-pt | 0.100 | de-nl | 0.300 | fr-nl | 0.300 |
| cn-ru | NaN | de-pt | 0.270 | fr-pt | 0.260 |
| cz-de | 0.240 | de-ru | NaN | fr-ru | NaN |
| cz-en | 0.250 | en-es | 0.420 | nl-pt | 0.150 |
| cz-es | 0.240 | en-fr | 0.320 | nl-ru | NaN |
| cz-fr | 0.170 | en-nl | 0.350 | pt-ru | NaN |

Table 2. Multifarm Result by Seals

The table shows that the worst results all happened in Chinese tasks, because the basic tool we use in all multifarm fashion is translate tool, we use both google translator and bing’s translator to initialize the label set before we calculate the WordNet similarity, edit-distance similarity and vector space similarity.

Because of the fact that information in each multifarm’s tasks is qualified, involuntarily, we got the limit on result, the highest F1 we got is 0.605 which is Czech ontology and English ontology ’s alignment on local machine.

2.3 instance matching

The result for Instance Matching 2013 is shown in Table 3.

As we can see from the table, we achieve high values for all measures in all five testcases, especially in testcase1 and 3. Furthermore, the official result shows that we win first prize in IM@OAEI2013. We confidently believe our algorithm, Link Propagation Algorithm is effective for instance matching. We owe our results to each module of the algorithm and further explain the results more specifically.

For testcase1, the Score Matching module exploits weighted average score, therefore avoiding emphasizing some particular information of instances. The reasons why we attain best performance in testcase1 also include little change in target ontology. In testcase2, with almost only link information, we needn’t employ last module Score Matching. Nevertheless, it achieves comparative performance, reflecting the power of

link information, in other words, the power of Link Flooding Algorithm. Though test-case 3, 4 and 5 have few initial links, we can find new matching pairs through Score Matching. Although only a few matching pairs are found, we can detect lots of new pairs by iteratively running Unique Subject Matching and One-left Object Matching.

| TestCase | Precision | Recall | F1-measure |
|------------|-----------|--------|------------|
| testcase01 | 1.00 | 1.00 | 1.00 |
| testcase02 | 0.95 | 0.99 | 0.97 |
| testcase03 | 0.96 | 0.99 | 0.98 |
| testcase04 | 0.94 | 0.98 | 0.96 |
| testcase05 | 0.93 | 0.99 | 0.96 |

Table 3. Instance Matching Result

3 General comments

3.1 Discussions on the way to improve the proposed system

We have got no split new method implemented during the benchmark task, and there also have much information in these tasks that we need to make them outcrop. And we have not run the RiMOM2013 on anatomy, conference, Library, etc. For anatomy, since many technical terms emerge as labels in ontologies, we should add some manually labelling step to generate the reference alignment result but the problem is how to determine a result pair is matched or not if we have not any biological knowledge. For multifarm, because the multifarm dataset is translated from conference collection, if we do the experiment on conference before multifarm, there may be a credible auxiliary information between each entity pair during the multifarm experiment.

3.2 Comments on the OAEI 2013 measures

The results show that in schema level matching, using description information gain the better matching result, by contrast, in instance level's matching, using linking information got the better result, because in instance level, the types of relationship between each entity is diverse, and in schema level is drab.

4 Conclusion

In this paper, we present the result of RiMOM2013 in OAEI 2013 Campaign. We participate in three tracks this year, including Benchmark, Multifarm and Instance Matching. We presented the architecture of RiMOM2013 framework and described specific techniques we used during this campaign. In our project, we design a new framework to do the ontology alignment task. We focus on the instance matching task and propose three new method in instance matching tasks. The results show that our project can both deal with multi-lingual ontology on schema level and do well on instance level, and this will be paid attention in the community.

5 Acknowledgement

The work is supported by NSFC (No. 61035004), NSFC-ANR(No. 61261130588), 863 High Technology Program (2011AA01A207), FP7-288342, and THU-NUS NExT Co-Lab.

References

1. Li, J., Tang, J., Li, Y., Luo, Q.: RiMOM: a dynamic multistrategy ontology alignment framework. *IEEE Trans. Knowl. Data Eng.* (2009) 1218–1232
2. Wang, Z., Zhang, X., Hou, L., Zhao, Y., Li, J., Qi, Y., Tang, J.: RiMOM results for oaei 2010. In: *OM'10*. (2010)
3. Melnik, S., Garcia-Molina, H., Rahm, E.: Similarity Flooding: A versatile graph matching algorithm and its application to schema matching. In: *ICDE'02*. (2002) 117–128
4. Meilicke, C., Garcia-Castro, R., Freitas, F., van Hage, W.R., Montiel-Ponsoda, E., de Azevedo, R.R., Stuckenschmidt, H., Svob-Zamazal, O., Svtok, V., Tamin, A., dos Santos, C.T., Wang, S.: Multifarm: A benchmark for multilingual ontology matching. *J. Web Sem.* (2012) 62–68
5. Wang, Z., Li, J., Wang, Z., Tang, J.: Cross-lingual knowledge linking across wiki knowledge bases. In: *WWW'12*. (2012) 459–468
6. Nguyen, K., Ichise, R., Le, B.: SLINT: a schema-independent linked data interlinking system. In: *OM'12*. (2012)